<u>EXHIBIT E</u>

VP/#66313480
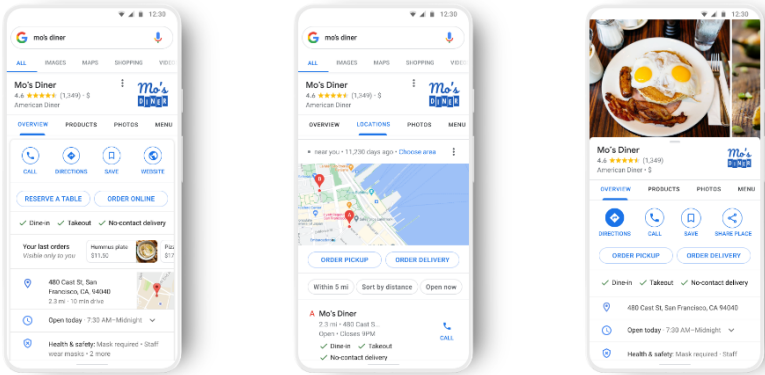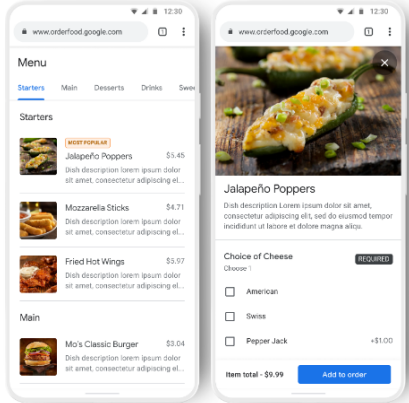
EXHIBIT E
(Page 1)

ELEMENT(S)	INDEPENDENT CLAIM 1: 11,038,090	ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD
Preamble, [1a]	A computer-implemented method for providing a searchable aggregated data structure for a networked application, the method comprising:	The Accused Product/Service is a computer-implemented method for providing a searchable aggregated data structure for a networked application for users to select their food choices as shown under the "Overview and Eligibility" on Google Food Developer Site:

EXHIBIT E
(Page 2)

ELEMENT(S)	INDEPENDENT CLAIM 1: 11,038,090	ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD
		Google Food allows a user to search food and menus for delivery from its member restaurants in a given location, including by common menu item, and with an "Order Delivery" button, as shown below for "chilean sea bass":

EXHIBIT E
(Page 3)

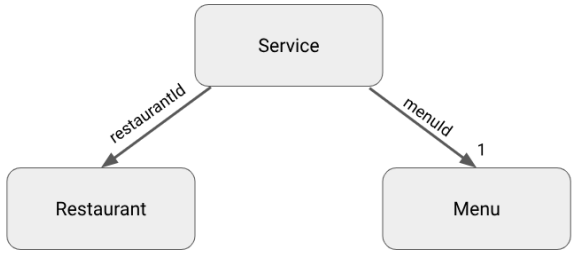| ELEMENT(S)	INDEPENDENT CLAIM 1: 11,038,090	ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD
		  Using Google Food's computer-implemented method for providing a searchable aggregated data structure for a networked application, a user can access a particular restaurant menu and order delivery directly through Google Food, as shown below for Portillo's in Deerfield, Illinois: See (https://developers.google.com/actions-center/verticals/ordering/e2e/overview)
[1b]	acquiring by a processor source data from a plurality of delivery service computers associated with a plurality of food or beverage delivery services over a communication network, the	Google Food acquires source data, by a processor, from a plurality of delivery service computers associated with a plurality of food or beverage delivery services over a communication network, the acquired source data being in a plurality of formats, where the acquired source data includes, for each of the plurality of food or beverage delivery services, data representing multiple source menu items, under the heading "Integration" step on the Google Food Developer Site (also shown below):

EXHIBIT E
(Page 4)

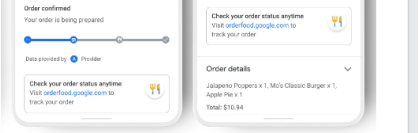| ELEMENT(S)	INDEPENDENT CLAIM 1: 11,038,090	ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD
	acquired source data being in a plurality of formats, where the acquired source data includes, for each of the plurality of food or beverage delivery services, data representing multiple source menu items, wherein said acquiring source data comprises one or more of:	**Integration** For your food business to connect with Google's Ordering End-to-End support, you must integrate with Google systems and APIs. There are three processes you need to implement to connect with Ordering End-to-End: 1. **Provide restaurant, menu, and service data feeds to Google.** The first step of the integration process is to create and host data feeds about your restaurant, menu, and service. These feeds provide details about restaurant name, location, service hours, menu items and sections, delivery areas, and more. Google ingests your data feeds and uses them to present your menu and services to users. You can update these feeds regularly and even incorporate incremental changes in real time. To implement these processes, the Ordering End-to-End integration has two main components: Inventory feeds and fulfillment actions. • **Inventory feeds** These feeds use a relational inventory schema to supply Google with up-to-date information about a restaurant, the services it provides, and the items in its menu. The structure of your Ordering End-to-End data feed is defined by the relational inventory schema. The Ordering End-to-End data feed consists of the following top-level entities: • `Restaurant` entities: Which restaurants you service. • `Service` entities: Timing, location, and conditions of your service. • `Menu` entities: Details of each restaurant's menu. The following diagram shows how `Service`, `Restaurant`, and `Menu` entities represent a single restaurant: *Figure 1: Overall relationship of Ordering End-to-End data feed entities: Service, Restaurant, and Menu.* **General guidelines** • **Restaurants per file**: Each data file should represent one restaurant with its related `Service` and `Menu` entities. Use filenames that can help you search a file for a restaurant. • **Data file format**: Data files need to be formatted in newline-separated JSON files (ndjson format). • **DateTime and Time values**: For properties that require a `DateTime` or `Time` value, use the formats specified in DateTime and Time formats. For example, `2017-05-01T06:30:00+05:30` for `DateTime` and `T08:08:00+05:30` for `Time`. • **IDs**: Use an `@id` property to identify all unique entities within an entity type. The maximum length is 300 characters. An `@id` is a unique identifier of the entity of that type, but across entities, IDs can overlap. For example, assume you define a `Service` entity with the `@id` property set to `a16`. You cannot create another `Service` entity with an `@id` of `a16`. However, you can use `a16` as the `@id` value of a `Menu` entity. • **ID Generation**: Keep your ids stable - do not use UUIDs or otherwise change/randomize ids between feed uploads. This allows easier support for entity-related issues. • **Null values**: Do not use the value `null` in place of objects. If an object is optional, you must omit it from your feed. See (https://developers.google.com/actions-center/verticals/ordering/e2e/overview)
[1c]	employing an application programming interface to interface with the plurality of	Google Food employs an application programming interface (API) to interface with the plurality of delivery service computers, as shown under the heading "Integration" step on Google Food Developer Site (also shown below):

EXHIBIT E
(Page 5)

ELEMENT(S)	INDEPENDENT CLAIM 1: 11,038,090	ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD
	delivery service computers, or	 Google Food also employs an Async Order Update API to interface with the plurality of delivery service computers for order updates and for users to check their order status, shown under the heading "Integration" step on Google Food Developer Site (also shown below): See (https://developers.google.com/actions-center/verticals/ordering/e2e/overview)
[1d]	scraping data from the plurality of delivery service computers;	Google Food scrape data from the plurality of delivery service computers by "ingesting" the data feeds, as shown below under the "Integration" step on Google Food Developer Site:

EXHIBIT E
(Page 6)

ELEMENT(S)	INDEPENDENT CLAIM 1: 11,038,090	ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD
		Update your data feeds After your data feeds are connected, Google checks for updates once each hour, but we only ingest all data feeds when the `marker.txt` or `sitemap.xml` files have been modified. We expect that you update your data feeds once a day to prevent stale inventory. To specify that the data feeds have been modified and are ready for batch ingestion, update the `last-modified` object metadata field of the `marker.txt` file (For GCP and S3) or the `last-modified` response header of the `sitemap.xml` file. Google uses these values to determine how fresh a data feed is. As the batch feed is being ingested, • New entities that don't exist in your current Ordering End-to-End inventory and don't have any errors would be inserted. • Entities already present in the inventory that don't have any errors on ingestion and either have a `dateModified` more recent than their current entry or in the case of not haveing a `dateModified` the feed ingestion start time is more recent than the current entry they would be updated, otherwise they would be marked as stale. • Entities that were part of a previous feed that are no longer included in the batch feed being processed would be deleted, provided there are no file level errors in the feed. The timestamp or the `last-modified` response header must be updated only after all of the data feed files are generated and updated. Limit the batch jobs that update your data feeds to run only once a day. Alternatively, have a gap of at least three hours between each batch job. If you don't take these steps, Google might fetch stale files. See (https://developers.google.com/actions-center/verticals/ordering/e2e/overview)
[1e]	mapping by the processor the acquired service data according to a predetermined data format to provide formatted data;	Google Food maps the acquired data according to a predetermined data format to provide formatted data. As shown below, Google Food maps the acquired service data (data feed) using relational inventory schema, under the heading "Integration" step on Google Food Developer Sit. • **Inventory feeds** These feeds use a relational inventory schema (/actions-center/verticals/ordering/e2e/reference/feeds/relational-inventory-schema) to supply Google with up-to-date information about a restaurant, the services it provides, and the items in its menu. Inventory feeds Inventory feeds are how a user discovers and orders from a desired restaurant. The relational inventory schema defines the structure of your Ordering End-to-End inventory. An Ordering End-to-End inventory feed consists of the following main entities: • `Restaurant` • `Service` • `Menu` The mapping of the ingested data is formatted for the Ordering End-to-End data feeds (Food Catalog Specification) using the JSON data schema.

EXHIBIT E
(Page 7)

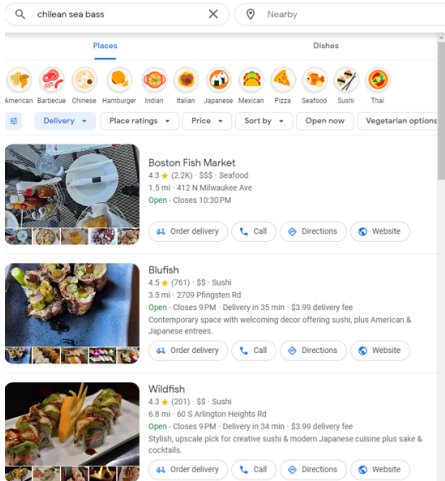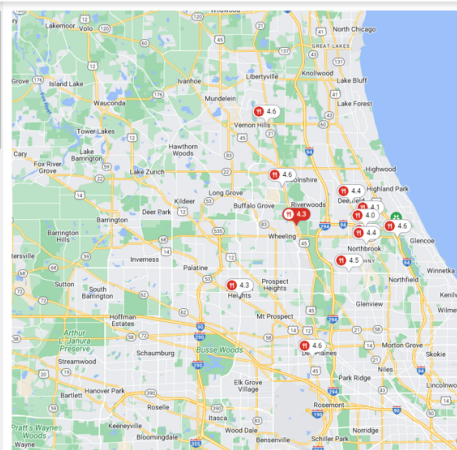| ELEMENT(S)	INDEPENDENT CLAIM 1: 11,038,090	ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD
		Relational inventory schema This page describes the format for the Ordering End-to-End data feeds (Food Catalog Specification) you provide to Google. For a machine-readable version of this information, you can download the JSON schema. **General requirements** Entities must be structured to be on one line per entity in the feeds (entities are separated by newline characters). For readability purposes, JSON examples on this page do not follow that structure. However, you must follow that structure when sending your feeds. For example, a menu entity must be structured like the following code: `{"@type": "Menu","name": "Coffee Shop A","@id": "1535"}` Each `Restaurant` entity may have two Service entities (one each for the `DELIVERY` and `TAKEOUT` service types). Each `Service` entity may only have one `Menu` entity. Any sub-entities can be reused across multiple restaurants. See (https://developers.google.com/actions-center/verticals/ordering/e2e/overview)
[1f]	linking by the processor the formatted data to common sources of the source menu items, each of the common sources being represented by source identification data uniquely identifying a respective one of the sources, such that at least one food or beverage delivery service is linked to each common source and its source menu items;	Google Food links the formatted data, by the processor, to common the formatted data to common sources of the source menu items, each of the common sources being represented by source identification data uniquely identifying a respective one of the sources, such that at least one food or beverage delivery service is linked to each common source and its source menu items, under the heading "Integration" step on Google Food Developer Site. Google Food links the formatted data by restaurant, service and menu items by using the "Inventory Feed" component of its Ordering End-to-End Integration system (part of the Google Food Developer Site). **Inventory feeds** Inventory feeds are how a user discovers and orders from a desired restaurant. The relational inventory schema defines the structure of your Ordering End-to-End inventory. An Ordering End-to-End inventory feed consists of the following main entities: • `Restaurant` • `Service` • `Menu` **Menu** The Menu entity describes a menu that a restaurant offers to its customers. A Menu entity must be defined within a Restaurant entity through the Restaurant's `menuId` property. Other objects that define the menu (such as description, image, add-ons, and nutrition information) include `MenuItem` and `MenuItemOffer`, as well as the optional objects `MenuSection`, `Availability`, and `MenuItemOption`. **Menu sections** You can use `MenuSection` objects to organize multiple `MenuItem` objects into logical categories. This approach is useful for restaurants that have multiple menus (like breakfast, lunch, and dinner). Add each menu as a separate `MenuSection`. In addition, Google Food offers merchant matching/linking services in order to ensure that inventory is shown to users in the right locations, under the Partner Portal of the Google Food Developer Site.

EXHIBIT E
(Page 8)

ELEMENT(S)	INDEPENDENT CLAIM 1: 11,038,090	ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD
		  See (https://developers.google.com/actions-center/verticals/ordering/e2e/partner-portal/inventory/merchant-matching)
[1g]	identifying by the processor common menu items among the source menu items in the formatted data, and, for each identified common menu item, associating the source menu items with a master menu item;	Google Food identifies, by the processor, common menu items among the source menu items in the formatted data, and, for each identified common menu item, associating the source menu items with a master menu item. For instance, Google Food allows a user to search food and menus for delivery from its member restaurants in a given location, including by common menu item, and with an "Order Delivery" button, as shown here for "chilean sea bass": Google Food also groups restaurants by restaurant type, as shown here for "hamburger":

EXHIBIT E
(Page 9)
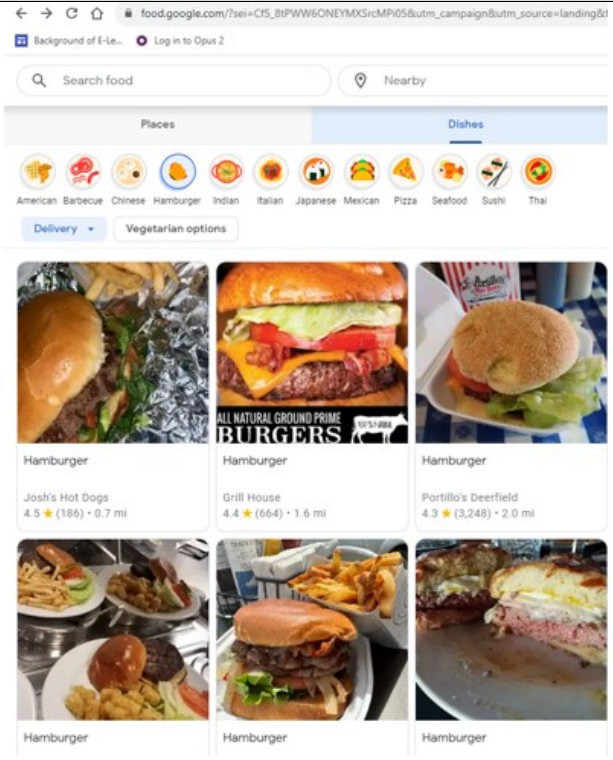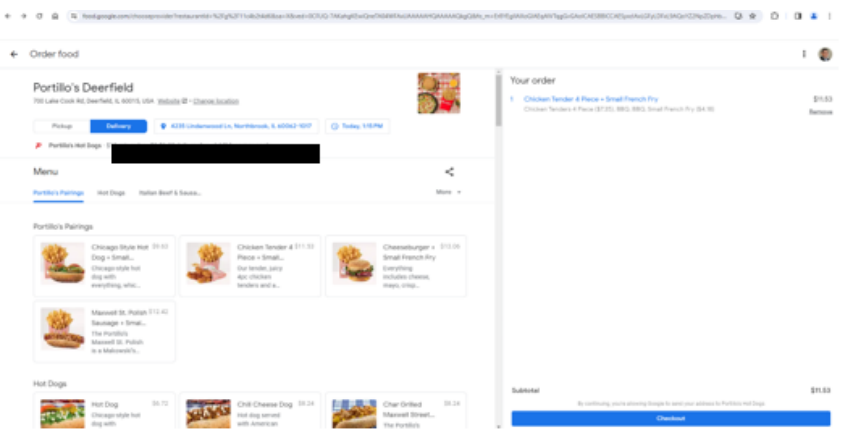
ELEMENT(S)	INDEPENDENT CLAIM 1: 11,038,090	ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD
		In addition, a user can access a particular restaurant menu and order delivery directly through Google Food, as shown here for Portillo's in Deerfield, Illinois: See (https://developers.google.com/actions-center/verticals/ordering/e2e/overview)
[1h]	combining by the processor the linked data and the master menu items into a	See above charts illustrating how Google Food combines the linked data and the master menu items into a master data set as shown in the "Inventory Feeds" and "Relational Inventory Schema" under the heading "Integration" step on Google Food Developer Site for its Ordering End-to-End Integration System (also shown below).

EXHIBIT E
(Page 10)

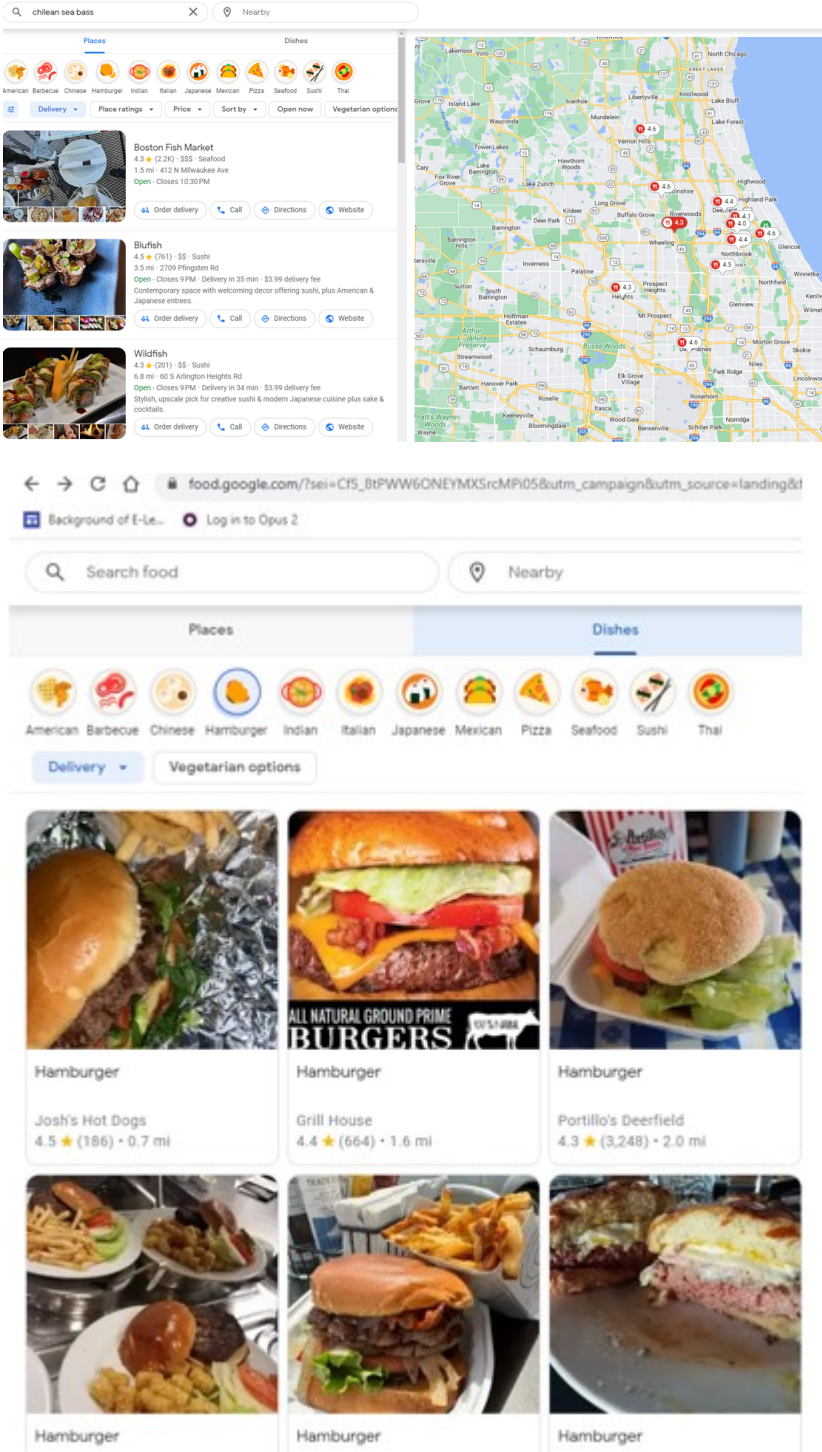| ELEMENT(S)	INDEPENDENT CLAIM 1: 11,038,090	ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD
	master data set; and	**Inventory feeds** Inventory feeds are how a user discovers and orders from a desired restaurant. The relational inventory schema defines the structure of your Ordering End-to-End inventory. An Ordering End-to-End inventory feed consists of the following main entities: • `Restaurant` • `Service` • `Menu` **Batch feeds** Google uses a daily batch feed of your inventory feeds to make Ordering End-to-End available to users. To keep your inventory up to date, you must update your batch feeds at least once a day for Google to fetch. It takes about two hours for your inventory to be updated by a batch. **Incremental Updates API** You can send time-sensitive updates of your inventory to Google. The Incremental Updates API lets you update and delete entities in your inventory in almost real time. Incremental updates are processed in no more than five minutes. This is primarily intended for updates that you can't foresee, such as emergency restaurant closures or removal of an out-of-stock item. If your change doesn't need to be reflected immediately, use the batch feeds instead. **Relational inventory schema** 🔖▾ Send feedback This page describes the format for the Ordering End-to-End data feeds (Food Catalog Specification) you provide to Google. For a machine-readable version of this information, you can download the JSON schema. **General requirements** Entities must be structured to be on one line per entity in the feeds (entities are separated by newline characters). For readability purposes, JSON examples on this page do not follow that structure. However, you must follow that structure when sending your feeds. For example, a menu entity must be structured like the following code: `{"@type": "Menu","name": "Coffee Shop A","@id": "1535"}` Each `Restaurant` entity may have two Service entities (one each for the `DELIVERY` and `TAKEOUT` service types). Each `Service` entity may only have one `Menu` entity. Any sub-entities can be reused across multiple restaurants. **Menu** The Menu entity describes a menu that a restaurant offers to its customers. A Menu entity must be defined within a Restaurant entity through the Restaurant's `menuId` property. Other objects that define the menu (such as description, image, add-ons, and nutrition information) include `MenuItem` and `MenuItemOffer`, as well as the optional objects `MenuSection`, `Availability`, and `MenuItemOption`. **Menu sections** You can use `MenuSection` objects to organize multiple `MenuItem` objects into logical categories. This approach is useful for restaurants that have multiple menus (like breakfast, lunch, and dinner). Add each menu as a separate `MenuSection`. Google Food, by combining the linked data and the master menu items into a master data set (a) allows a user to search food and menus for delivery from its member restaurants in a given location, including by common menu item, and with an "Order Delivery" button for "chilean sea bass" and/or (b) groups restaurants by restaurant type, as shown below for "hamburger":

VP/#66313480

EXHIBIT E
(Page 11)

ELEMENT(S)	INDEPENDENT CLAIM 1: 11,038,090	ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD
[1i]	importing by the processor the master data set and the source identification data	See above charts illustrating how Google Food imports the master data set and the source identification data into the searchable aggregated data structure, under the "Inventory Feeds" and "Relational Inventory Schema" sections of the "Integration"

EXHIBIT E
(Page 12)
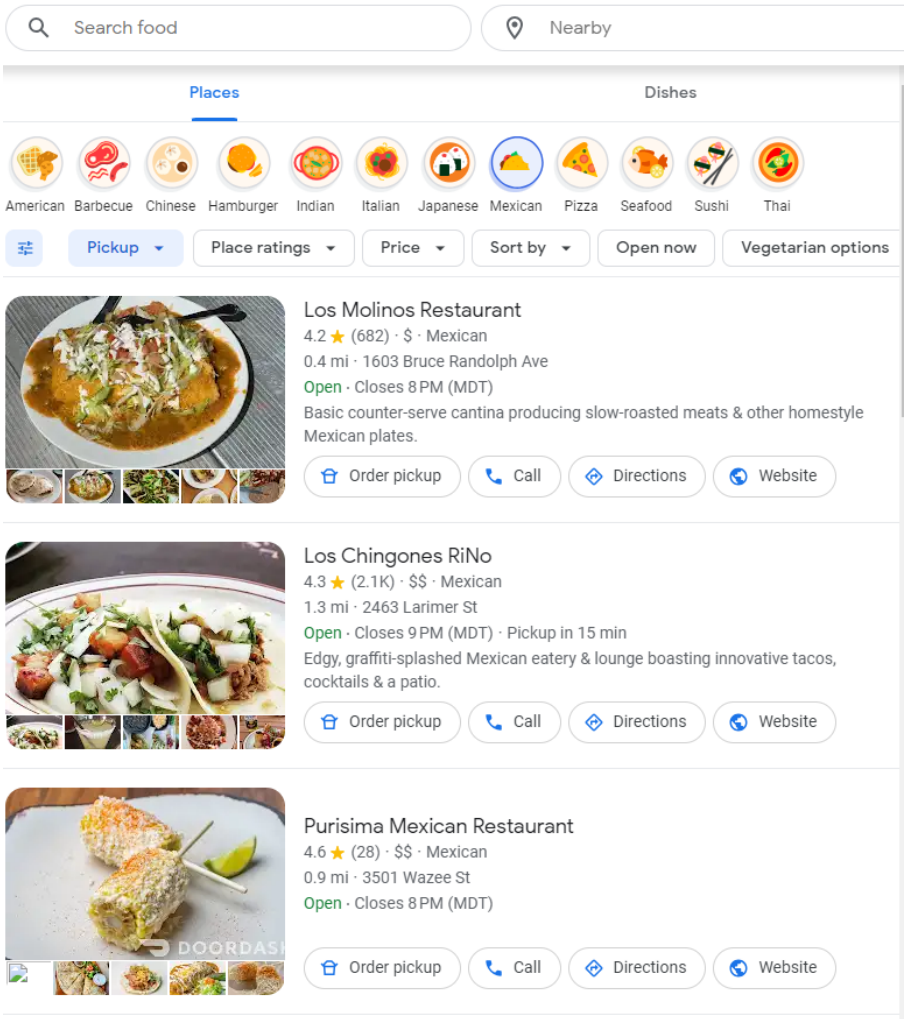
ELEMENT(S)	INDEPENDENT CLAIM 1: 11,038,090	ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD
	into the searchable aggregated data structure.	step on Google Food Developer Site for its Ordering End-to-End Integration System (also shown below).

EXHIBIT E
(Page 13)

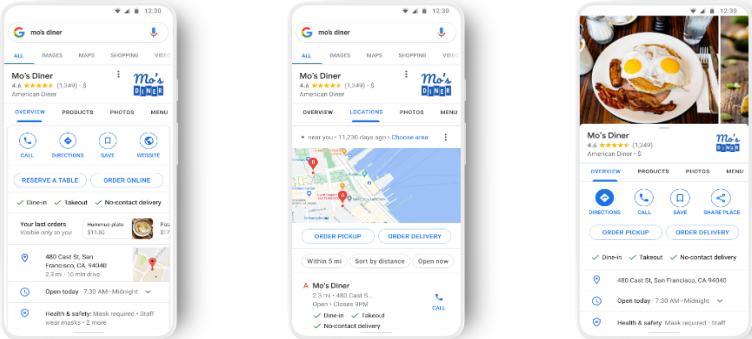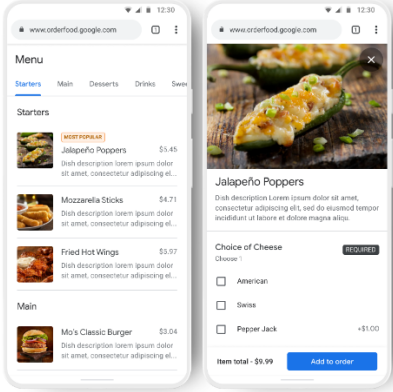| ELEMENT(S)	INDEPENDENT CLAIM 1: 11,038,090	ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD

ELEMENT(S)	DEPENDENT CLAIM 3: 11,038,090	ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD
[3a]	The method of claim 1, wherein said mapping the acquired data comprises: aliasing fields of the acquired data from formats used by the delivery service computers to respective fields of the predetermined data format:	See above, claim chart for independent claim 1 illustrating Google Food aliasing fields of the acquired data from formats used by the plurality of delivery service computers to respective fields of the predetermined data format. under the "Integration" step on Google Food Developer Site. In addition Google Food meet this limitation, since the restaurant identifier data comprise one or more of restaurant name data, restaurant location data, or restaurant identification code, as shown below.

Restaurant data

Restaurant (required)

A required entity to implement. Describes a restaurant.

The following table lists the properties for the `Restaurant` type:

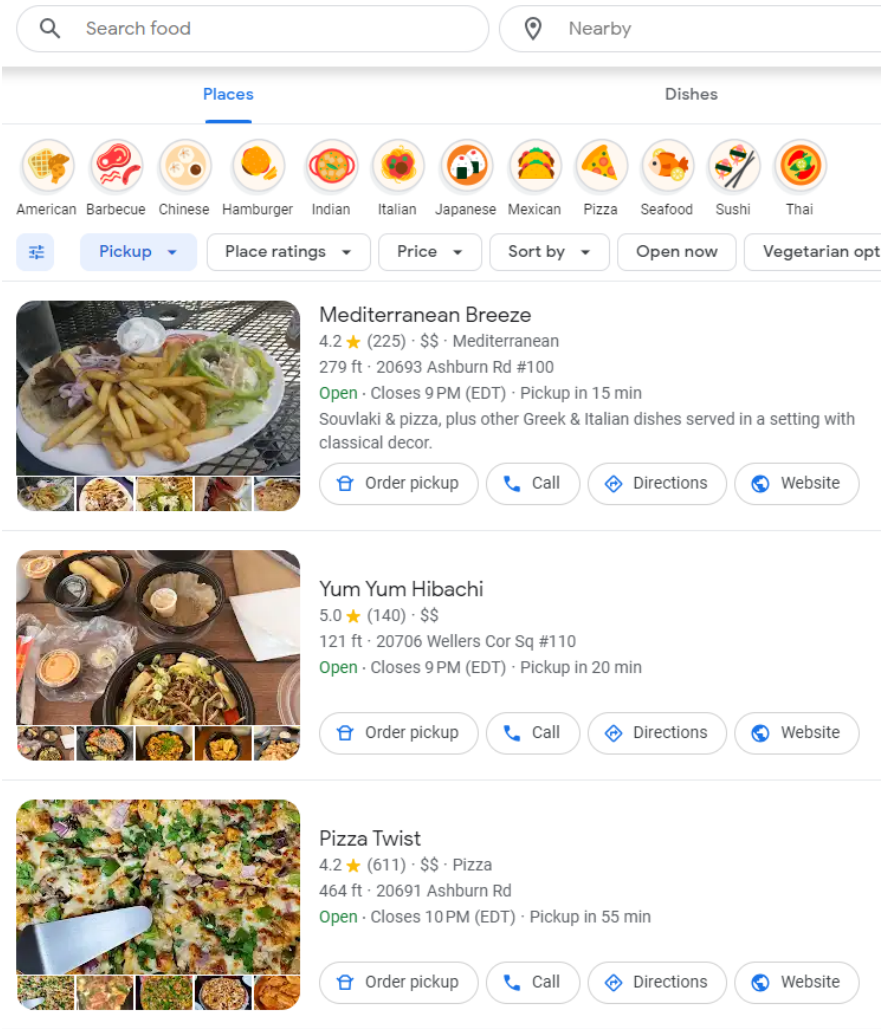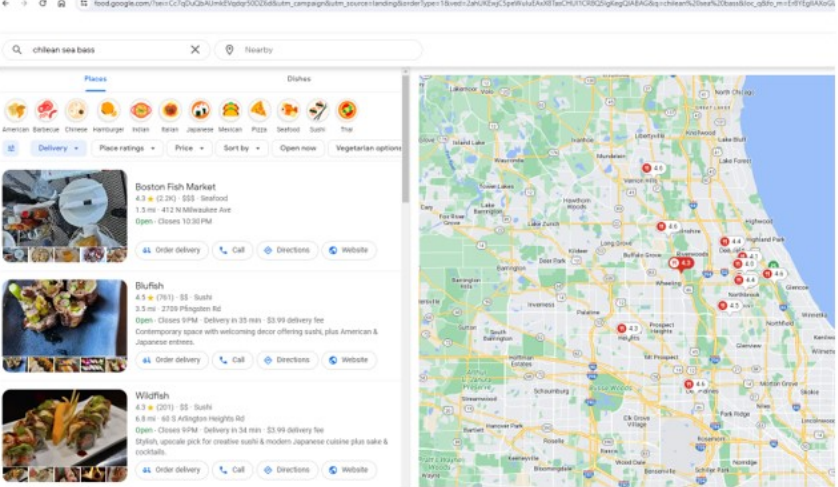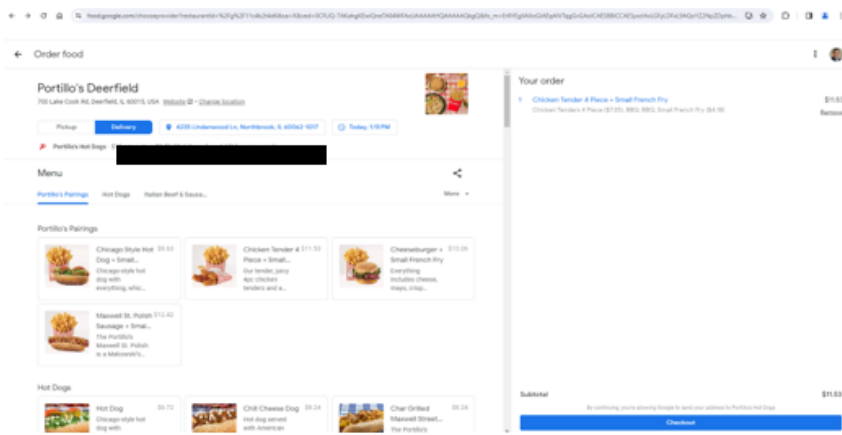| Property	Type	Description
@type	Const	Required.
		Value: Restaurant
@id	String	Required.
		A unique identifier of the restaurant or delivery provider.
		Example: restaurant_1
name	String	Required.
		Name of the restaurant.
		Example: Foo
description	String	A description of the restaurant.
		Example: Best seafood in town
url	Url	The URL that represents the restaurant. The restaurant domain is preferred over the aggregator domain.
		Example: http://www.provider.com/somerestaurant

EXHIBIT E
(Page 15)

ELEMENT(S)	DEPENDENT CLAIM 3: 11,038,090	ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD
		sameAs — *Url* — The official website for the restaurant. Example: `http://www.provider2.com/somerestaurant`
		telephone — *String* — Telephone number of the restaurant. Example: `+12345665898`
		streetAddress — *String* — Required. The street address of the restaurant. Example: `12345 Bar Avenu`
		addressLocality — *String* — Required. The locality or city. Example: `San Francisco`
		addressRegion — *String* — Required. The region or state. Example: `CA`
		postalCode — *String* — Required. The postal code. Example: `94124`
		addressCountry — *String* — Required. Two-letter ISO 3166-1 alpha-2 country code. Example: `US`
		latitude — *Number* — Latitude in degrees. Values are restricted to the range [[-90, 90]]. The precision should be at least 5 decimal places. Example: `35.7392607`
		longitude — *Number* — Longitude in degrees. Values are restricted to the range [[-180, 180]]. The precision should be at least 5 decimal places. Example: `-120.3895522`

Sample code from Google Food Developer Site implemented by Google Food to meet this limitation:

Example

```
{
  "@type": "Restaurant",
  "@id": "10824",
  "name": "Pronto Wood Fired Pizzeria",
  "url": "https://www.provider.com/pronto-wood-fired-pizzeria",
  "telephone": "+16503659978",
  "streetAddress": "2560 El Camino Real",
  "addressLocality": "Palo Alto",
  "addressRegion": "CA",
  "postalCode": "94061",
  "addressCountry": "US",
  "latitude": 37.472842,
  "longitude": -122.217144
}
```

VP/#66313480

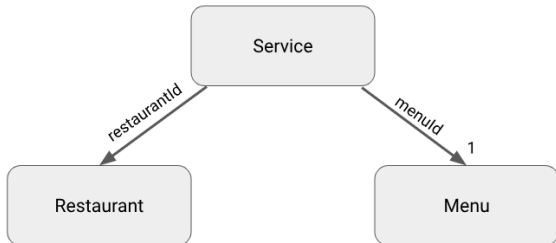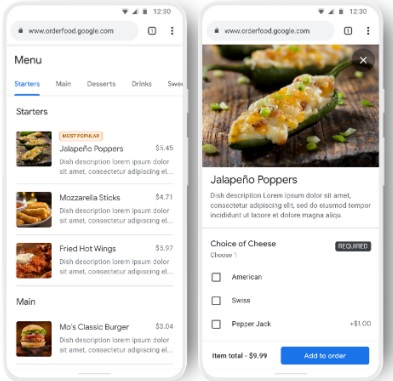| ELEMENT(S)	DEPENDENT CLAIM 4: 11,037,090	ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD
[4a]	The method of claim 1, wherein the source identification data comprise one or more of name data, location data, or an identification code.	See above, claim chart for independent claim 1, where the source identification data comprise one or more of name data, location data, or an identification code.
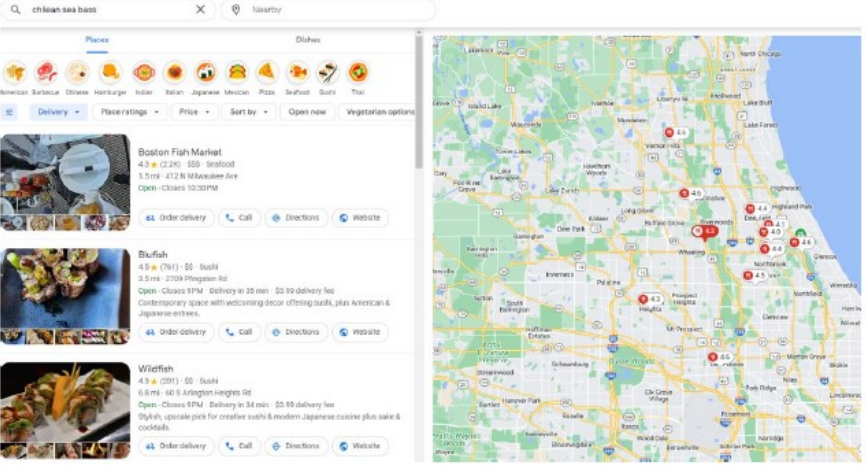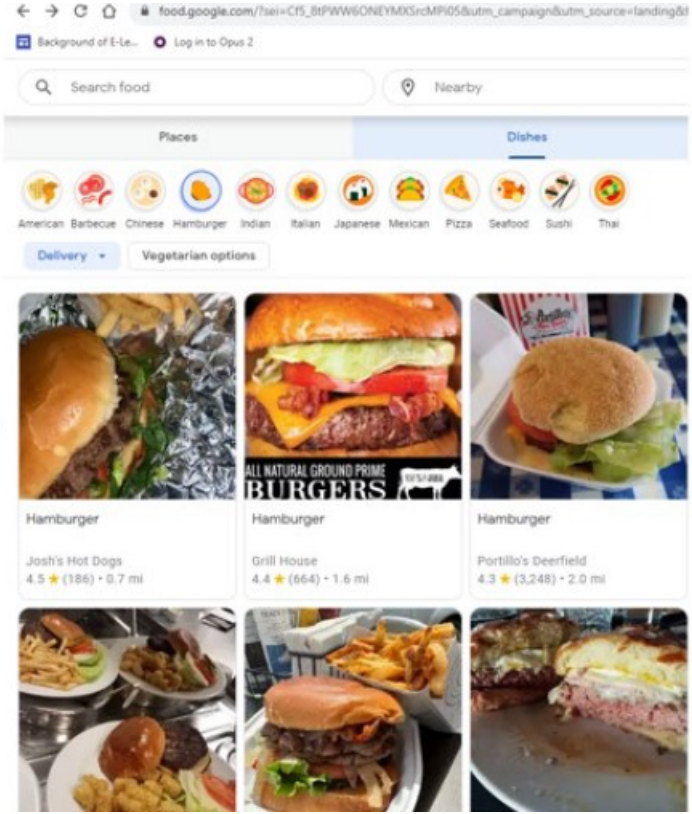
ELEMENT(S)	DEPENDENT CLAIM 5: 11,037,090	ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD
[5a]	The method of claim 4, wherein said linking comprises: analyzing the source identification data to determine the common sources; and linking the formatted data associated with like common sources to a master data object.	See above, claim chart for independent claim 1, showing how Google Food links by analyzing the source identification data to determine the common sources; and linking the formatted data associated with like common sources to a master data object, a under the "Integration" step on Google Food Developer Site for its Ordering End-to-End Integration System, as shown below.

ELEMENT(S)	INDEPENDENT CLAIM 12: 11,038,090	ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD
Preamble, [12a]	A system for providing an interactive food or beverage ordering service accessible by a user computing device, the system comprising:	The Accused Product/Service is a system, an Ordering End-to-End Integration System as shown on the Google Food Developer Site, for providing an interactive food or beverage ordering service accessible by a user computing device, as shown below

ELEMENT(S)	INDEPENDENT CLAIM 12: 11,038,090	ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD
		  The system allows a user to search food, beverages and menus for delivery from its member restaurants in a given location, including by common menu item, and with an "Order Delivery" button, as shown below for "chilean sea bass":

ELEMENT(S)	INDEPENDENT CLAIM 12: 11,038,090	ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD
		Using Google Food's system for providing an interactive food or beverage ordering service accessible by a user computing device, a user can access a particular restaurant menu and order delivery directly through Google Food, as shown below for Portillo's in Deerfield, Illinois: See (https://developers.google.com/actions-center/verticals/ordering/e2e/overview)
[12b]	a data acquisition and processing module comprising a processor, memory, and computer-readable instructions stored on a non-transitory medium that are executable by the processor to acquire source data from a	Google Food comprises a data acquisition and processing module (which evidently includes a processor, memory accessible to the processor, and a set of computer-readable instructions stored on a non-transitory medium that are executable by the processor) in order to acquire source data from a plurality of delivery service computers associated with a plurality of food or beverage delivery services and provide a master data set of formatted data, wherein the master data set includes, for each of the plurality of food or beverage delivery services, data representing multiple menu items linked to identification data uniquely identifying sources of the menu items delivered by the plurality of food or beverage delivery services as shown on Google Food Developer Site:

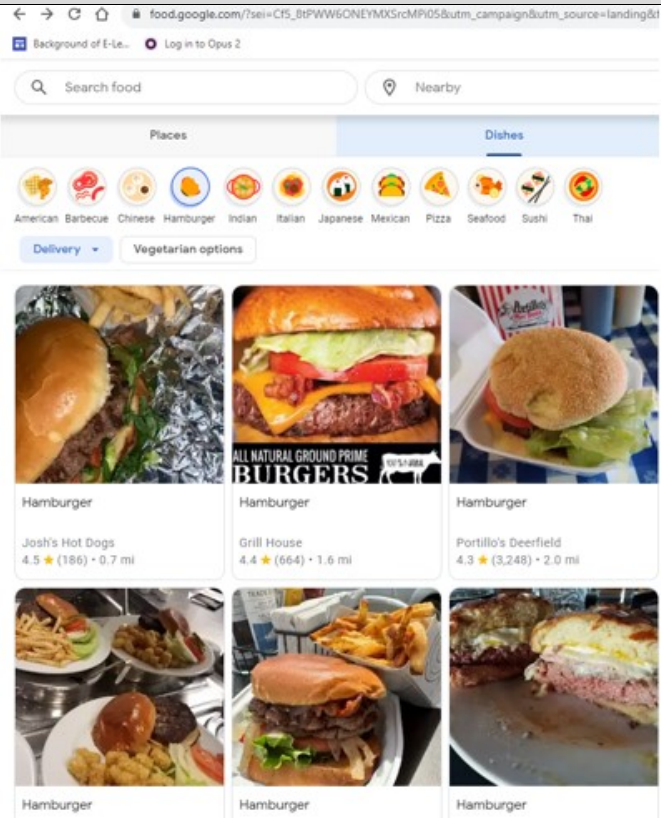| ELEMENT(S)	INDEPENDENT CLAIM 12: 11,038,090	ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD
	plurality of delivery service computers associated with a plurality of food or beverage delivery services and provide a master data set of formatted data, wherein the master data set includes, for each of the plurality of food or beverage delivery services, data representing multiple menu items linked to identification data uniquely identifying sources of the menu items delivered by the plurality of food or beverage delivery services; and	**Integration** For your food business to connect with Google's Ordering End-to-End support, you must integrate with Google systems and APIs. There are three processes you need to implement to connect with Ordering End-to-End: 1. **Provide restaurant, menu, and service data feeds to Google.** The first step of the integration process is to create and host data feeds about your restaurant, menu, and service. These feeds provide details about restaurant name, location, service hours, menu items and sections, delivery areas, and more. Google ingests your data feeds and uses them to present your menu and services to users. You can update these feeds regularly and even incorporate incremental changes in real time. **Inventory feeds** Inventory feeds are how a user discovers and orders from a desired restaurant. The relational inventory schema defines the structure of your Ordering End-to-End inventory. An Ordering End-to-End inventory feed consists of the following main entities: • Restaurant • Service • Menu The structure of your Ordering End-to-End data feed is defined by the relational inventory schema. The Ordering End-to-End data feed consists of the following top-level entities: • Restaurant entities: Which restaurants you service. • Service entities: Timing, location, and conditions of your service. • Menu entities: Details of each restaurant's menu. The following diagram shows how Service, Restaurant, and Menu entities represent a single restaurant: **Figure 1:** Overall relationship of Ordering End-to-End data feed entities: Service, Restaurant, and Menu. **The user selects their food choices** The user can browse through multiple menu sections and options to select their food choices, and add items to their cart along the way. They can customize their order with add-ons or special instructions, repeat previous orders, and add suggested or popular items. Then, they make their food choices and add them to their cart. Google Food provides a master data set of formatted data, wherein the master data set includes, for each one of the plurality of food delivery computers, data representing multiple menu items provided by multiple restaurants. For instance, Google Food allows a user to search food and menus for delivery from its member

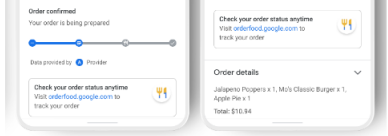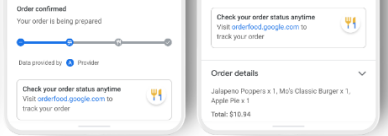| ELEMENT(S)	INDEPENDENT CLAIM 12: 11,038,090	ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD
		restaurants in a given location, including by common menu item, and with an "Order Delivery" button, as shown here for "chilean sea bass": Google Food also groups restaurants, within its master data set, by restaurant type, as shown here for "hamburger": See (https://developers.google.com/actions-center/verticals/ordering/e2e/overview)
[12c]	a website database accessible to the processor and	Google Food comprises a website database accessible to the processor and configured to receive updated data from the master data set, the master data set

ELEMENT(S)	INDEPENDENT CLAIM 12: 11,038,090	ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD
	configured to receive updated data from the master data set, the master data set representing the multiple menu items provided by one or more of the plurality of food or beverage delivery services;	representing the multiple menu items provided by one or more of the plurality of food or beverage delivery services. As shown below, the first step of the integration process is to create and host data feeds about restaurants, menus, and services. These data feeds provide details about restaurant name, location, service hours, menu items and sections, delivery areas, and more. Next, Google Food ingests these data feeds and uses them to present menu and services to end-users. These data feeds are updated regularly and even incremental changes to the data feed is incorporated in real time within the Google Food system. Further, as noted under the "Integration" step on Google Food Developer Site (and shown below), Google food is configured with specific sub-systems and modules such as Inventory Feeds, Batch Feeds and Incremental Updates API for receiving the most recent data from the master data set time-sensitive updates of the data. Google Food provides master data set representing the multiple menu items provided by each one of the plurality of food or beverage delivery services for the multiple restaurants for end-to-end ordering, as shown below:

ELEMENT(S)	INDEPENDENT CLAIM 12: 11,038,090	ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD
		  See (https://developers.google.com/actions-center/verticals/ordering/e2e/overview)
[12d]	wherein the data acquisition and processing module further comprises a plurality of modules in the form of a computer-readable instructions stored on a non-transitory medium that are executable by the processor including:	Google Food comprises a data acquisition and processing module (which evidently comprises a plurality of modules in the form of a computer-readable instructions stored on a non-transitory medium that are executable by the processor) in order to acquire and map the source data from a plurality of delivery service computers. See (https://developers.google.com/actions-center/verticals/ordering/e2e/overview)
[12e]	an application programming interface configured to interface the data acquisition and processing module with the	Google Food comprises an application programming interface (API) to interface the data acquisition and processing module with the plurality of delivery service computers, as shown under the heading "Integration" step on Google Food Developer Site (also shown below):

ELEMENT(S)	INDEPENDENT CLAIM 12: 11,038,090	ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD
	plurality of delivery service computers; [DON'T SEE THE "OR" IN THIS LIMITATION– IT IS PROBABLY IN THE 683- Will check]	 Google Food also employs an Async Order Update API to interface with the plurality of delivery service computers for order updates and for users to check their order status, shown under the heading "Integration" step on Google Food Developer Site (also shown below):
[12f]	an extraction module configured to extract the source data from the plurality of delivery service computers as raw files by scraping data from one or more of the plurality of delivery service computers;	Google Food comprises an extraction module configured to extract the source data from the plurality of delivery service computers as raw files by scraping data from one or more of the plurality of delivery service computers using JSON data feed schemes for its Ordering and end-to-end integration system, as shown in the "Integration," "Inventory Feeds" and "Relational Inventory Schema" sections under the "Integration" step on the Google Food Developer Site.

ELEMENT(S)	INDEPENDENT CLAIM 12: 11,038,090	ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD
		Relational inventory schema Send feedback This page describes the format for the Ordering End-to-End data feeds (Food Catalog Specification) you provide to Google. For a machine-readable version of this information, you can download the JSON schema. **General requirements** Entities must be structured to be on one line per entity in the feeds (entities are separated by newline characters). For readability purposes, JSON examples on this page do not follow that structure. However, you must follow that structure when sending your feeds. For example, a menu entity must be structured like the following code: `{"@type": "Menu","name": "Coffee Shop A","@id": "1535"}` Each `Restaurant` entity may have two Service entities (one each for the `DELIVERY` and `TAKEOUT` service types). Each `Service` entity may only have one `Menu` entity. Any sub-entities can be reused across multiple restaurants. **Generate client libraries** Send feedback A machine-readable version of the Ordering End-to-End data feed and fulfillment API definitions are made available to generate client source code and validate the structure of JSON data. This allows you to spend more time in developing application capabilities and business logic required for the integration. In this example, we use quicktype ⧉ CLI to generate an easy-to-use client library. **Download the JSON Schemas** These machine-readable versions of data feeds and APIs are needed for code generation and validation. • **Ordering End-to-End data feeds** • **Checkout and Submit Order API** • **Real-time updates** See (https://developers.google.com/actions-center/verticals/ordering/e2e/overview) See (https://developers.google.com/actions-center/verticals/ordering/e2e/reference/feeds/relational-inventory-schema) See (https://developers.google.com/actions-center/verticals/ordering/e2e/reference/tools/generate-client-libraries)
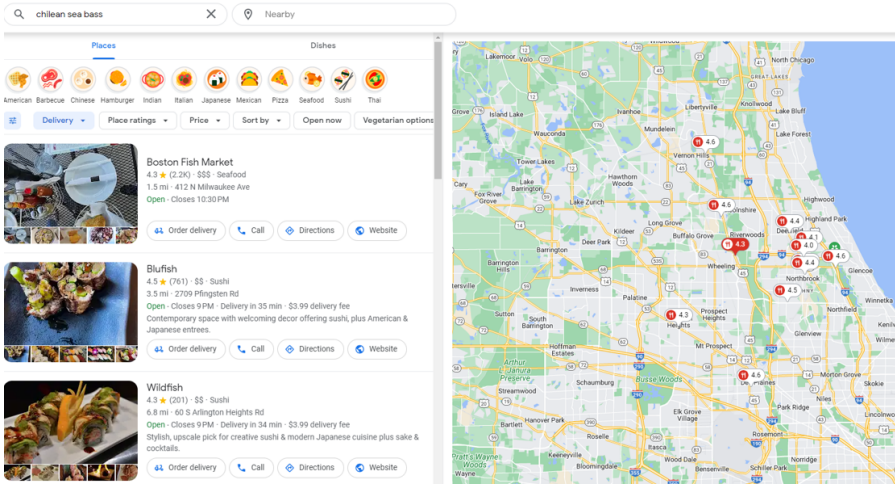
[12g]	a mapping module configured to convert the raw files to a standardized format to provide formatted data;	Google Food comprises a mapping module configured to convert the raw files to a standardized format to provide formatted data. under the "Integration" step on Google Food Developer Site. Google Food maps the acquired raw files using relational inventory schema, as shown below:

Inventory feeds

These feeds use a relational inventory schema
(/actions-center/verticals/ordering/e2e/reference/feeds/relational-inventory-schema) to supply Google with up-to-date information about a restaurant, the services it provides, and the items in its menu.

Inventory feeds

Inventory feeds are how a user discovers and orders from a desired restaurant. The relational inventory schema defines the structure of your Ordering End-to-End inventory. An Ordering End-to-End inventory feed consists of the following main entities:
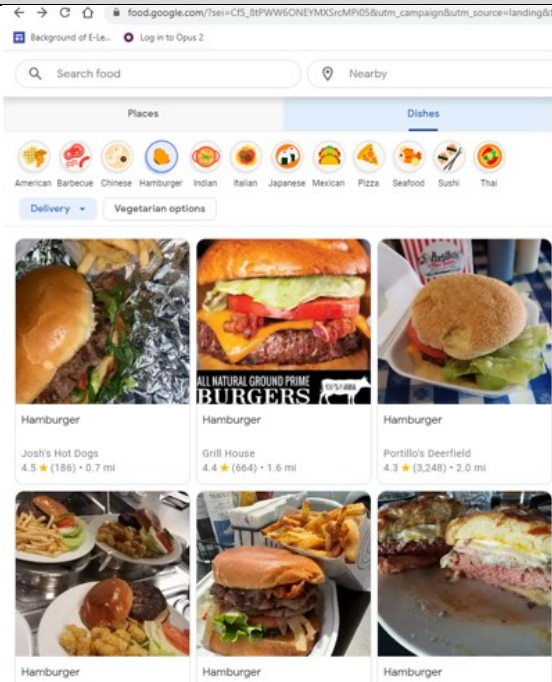
- `Restaurant`
- `Service`
- `Menu`

The mapping module is configured to convert the raw files to a standardized format to provide formatted data. The data is formatted for the Ordering End-to-End data feeds (Food Catalog Specification) using the JSON data schema.

Relational inventory schema   📑 ▾   Send feedback

This page describes the format for the Ordering End-to-End data feeds (Food Catalog Specification) you provide to Google. For a machine-readable version of this information, you can download the JSON schema.

General requirements

Entities must be structured to be on one line per entity in the feeds (entities are separated by newline characters). For readability purposes, JSON examples on this page do not follow that structure. However, you must follow that structure when sending your feeds. For example, a menu entity must be structured like the following code:

```
{"@type": "Menu","name": "Coffee Shop A","@id": "1535"}
```

Each `Restaurant` entity may have two Service entities (one each for the `DELIVERY` and `TAKEOUT` service types). Each `Service` entity may only have one `Menu` entity.

Any sub-entities can be reused across multiple restaurants.

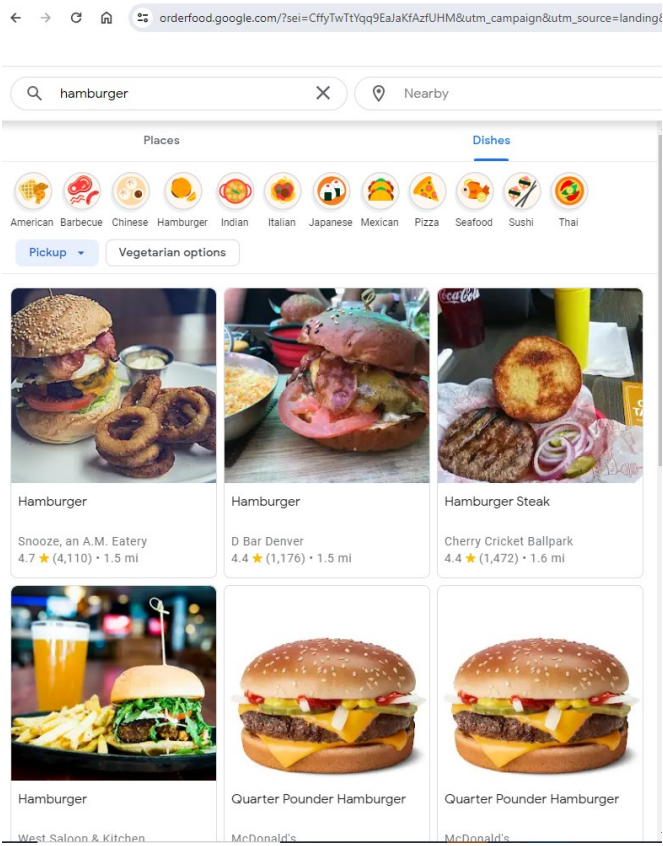| ELEMENT(S)	INDEPENDENT CLAIM 12: 11,038,090	ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD
		Restaurant data **Restaurant (required)** A required entity to implement. Describes a restaurant. The following table lists the properties for the `Restaurant` type: | Property	Type	Description
@type	Const	Required. Value: `Restaurant`
@id	String	Required. A unique identifier of the restaurant or delivery provider. Example: `restaurant_1`
name	String	Required. Name of the restaurant. Example: `Foo`
description	String	A description of the restaurant. Example: `Best seafood in town`
url	Url	The URL that represents the restaurant. The restaurant domain is preferred over the aggregator doma Example: `http://www.provider.com/somerestaurant` | See (https://developers.google.com/actions-center/verticals/ordering/e2e/overview)
[12h]	a linking module configured to perform record linkage on the formatted data according to the identification data that uniquely identifies sources; and	Google Food comprises a linking module configured to perform record linkage on the formatted data according to the identification data that uniquely identifies sources under the "Integration" step on Google Food Developer Site for its Ordering End-to-End Integration System. Google Food links the formatted data by restaurant, service and menu items by using the "Inventory Feed" component of its Ordering End-to-End Integration system, as shown below from the Google Food Developer Site. **Inventory feeds** Inventory feeds are how a user discovers and orders from a desired restaurant. The relational inventory schema defines the structure of your Ordering End-to-End inventory. An Ordering End-to-End inventory feed consists of the following main entities: • `Restaurant` • `Service` • `Menu` **Menu** The Menu entity describes a menu that a restaurant offers to its customers. A Menu entity must be defined within a Restaurant entity through the Restaurant's `menuId` property. Other objects that define the menu (such as description, image, add-ons, and nutrition information) include `MenuItem` and `MenuItemOffer`, as well as the optional objects `MenuSection`, `Availability`, and `MenuItemOption`. **Menu sections** You can use `MenuSection` objects to organize multiple `MenuItem` objects into logical categories. This approach is useful for restaurants that have multiple menus (like breakfast, lunch, and dinner). Add each menu as a separate `MenuSection`. In addition, Google Food offers merchant matching/linking services in order to ensure that inventory is shown to users in the right locations, under the Partner Portal of the Google Food Developer Site.:

ELEMENT(S)	INDEPENDENT CLAIM 12: 11,038,090	ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD
		Ordering End-to-End Actions Center Merchant Matching and Testing Send feedback **Match your inventory** The Inventory Viewer is your entrypoint to view all of the inventory that has been submitted to the Actions Center along with its matching status. **What is matching?** Matching is how you ensure that inventory is shown to users in the right locations. A Match is the connection between the Match Input, information you provide that are used for matching (typically information like merchant name, address, geo coordinate, phone number, and url), and the Match Output (typically a Google listing such as a business profile). All inventory that shares the same Match Input also share the same Match Output. This means that when a match is changed for one piece of inventory, all other inventory that uses that Match Input is updated as well. When changing the match the Actions Center lists all the inventory that would be impacted by the change. Within the Inventory Viewer the Matched column displays the match status (yes or no) on each piece of inventory. ★ **Note:** You can use filters, such as an entity ID or match status, to locate inventory. See (https://developers.google.com/actions-center/verticals/ordering/e2e/partner-portal/inventory/merchant-matching)
[12i]	a menu combining module configured to combine multiple source menus from linked sources into the master data set.	Google Food comprises a menu combining module configured to combine multiple source menus from linked sources into the master data set as shown in the "Inventory Feeds" and "Relational Inventory Schema" sections under the "Integration" step on Google Food Developer Site for its Ordering End-to-End Integration System, as shown in the above-charts and also illustrated below. **Inventory feeds** Inventory feeds are how a user discovers and orders from a desired restaurant. The relational inventory schema defines the structure of your Ordering End-to-End inventory. An Ordering End-to-End inventory feed consists of the following main entities: • Restaurant • Service • Menu **Batch feeds** Google uses a daily batch feed of your inventory feeds to make Ordering End-to-End available to users. To keep your inventory up to date, you must update your batch feeds at least once a day for Google to fetch. It takes about two hours for your inventory to be updated by a batch. **Incremental Updates API** You can send time-sensitive updates of your inventory to Google. The Incremental Updates API lets you update and delete entities in your inventory in almost real time. Incremental updates are processed in no more than five minutes. This is primarily intended for updates that you can't foresee, such as emergency restaurant closures or removal of an out-of-stock item. If your change doesn't need to be reflected immediately, use the batch feeds instead.

ELEMENT(S)	INDEPENDENT CLAIM 12: 11,038,090	ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD
		Relational inventory schema 🔖 ▾ Send feedback This page describes the format for the Ordering End-to-End data feeds (Food Catalog Specification) you provide to Google. For a machine-readable version of this information, you can download the JSON schema. **General requirements** Entities must be structured to be on one line per entity in the feeds (entities are separated by newline characters). For readability purposes, JSON examples on this page do not follow that structure. However, you must follow that structure when sending your feeds. For example, a menu entity must be structured like the following code: `{"@type": "Menu","name": "Coffee Shop A","@id": "1535"}` Each `Restaurant` entity may have two Service entities (one each for the `DELIVERY` and `TAKEOUT` service types). Each `Service` entity may only have one `Menu` entity. Any sub-entities can be reused across multiple restaurants. **Menu** The Menu entity describes a menu that a restaurant offers to its customers. A Menu entity must be defined within a Restaurant entity through the Restaurant's `menuId` property. Other objects that define the menu (such as description, image, add-ons, and nutrition information) include `MenuItem` and `MenuItemOffer`, as well as the optional objects `MenuSection`, `Availability`, and `MenuItemOption`. **Menu sections** You can use `MenuSection` objects to organize multiple `MenuItem` objects into logical categories. This approach is useful for restaurants that have multiple menus (like breakfast, lunch, and dinner). Add each menu as a separate `MenuSection`. For instance, the menu combining module in Google Food is configured to combine multiple source menus from linked sources into the master data set — to combine multiple source menus from linked restaurants into the master data set — in order to (a) allow a user to search food and menus for delivery from its member restaurants in a given location, including by common menu item, and with an "Order Delivery" button for "chilean sea bass" and/or (b) group restaurants by restaurant type, as shown below for "hamburger":

ELEMENT(S)	INDEPENDENT CLAIM 12: 11,038,090	ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD

ELEMENT(S)	DEPENDENT CLAIM 13: 11,037,090	ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD
[13a]	The system of claim 12, wherein the website database comprises the master data set in a searchable format.	See above, claim chart for independent claim 1, showing the website database comprises the master data set in a searchable format, as shown below.

ELEMENT(S)	DEPENDENT CLAIM 14: 11,037,090	ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD
[14a]	The system of claim 12, further comprising: a data warehouse configured to store the provided master data set in a searchable format; wherein the data warehouse is accessible by the website database to receive data.	See above, claim chart for independent claim 1, showing data acquisition which is, upon information and belief, stored in a data warehouse (data storage) configured to store the provided master data set in a searchable format; wherein the data warehouse is accessible by the website database to receive data. The data is searchable as shown below.

INDEPENDENT CLAIM 15: 11,037,090	ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD
An apparatus for providing a searchable aggregated data structure for a networked application, the apparatus comprising:	See, above claim charts for independent claims 1 and 12, showing Google Food constitute an apparatus for providing a searchable aggregated data structure for a networked application. See, e.g., elements [1a] and [1i] in claim chart 1.
a processor;	See, above claim charts for independent claims 1 and 12. See, e.g., elements [1b], [1e] – [1i] and [12b] and [12c] in claim charts 1 and 12.
a memory accessible by the processor; and	See, above claim charts for independent claims 1 and 12. See, e.g., element [12b] in claim chart 12.
a set of computer-readable instructions stored on a non-transitory medium accessible by the processor, the instructions being executable by the processor to perform a method comprising:	See, above claim charts for independent claims 1 and 12. See, e.g., elements [12b] and [12d] in claim chart 12.
acquiring source data from a plurality of delivery service computers associated with a plurality of food or beverage delivery services over a communication network, the acquired source data being in a plurality of formats, where the acquired source data includes, for each of the plurality of food or beverage delivery services, data representing multiple source menu items, and wherein said acquiring source data comprises one or more of:	See, above claim charts for independent claims 1 and 12. See, e.g., elements [1b], [12b] and [12f] in claim charts 1 and 12.
employing an application programming	See, above claim charts for independent claims 1 and 12. See, e.g., elements [1c] and [12e] in claim charts 1 and 12.

VP/#66313480

INDEPENDENT CLAIM 15: 11,037,090	**ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD**
interface to interface with the plurality of delivery service computers, or	
scraping data from the plurality of delivery service computers;	See, above claim charts for independent claims 1 and 12. See, e.g., elements [1d] and [12f] in claim charts 1 and 12.
mapping the acquired source data according to a predetermined data format to provide formatted data;	See, above claim charts for independent claims 1 and 12. See, e.g., elements [1e], [3a] and [12g] in claim charts 1, 3 and 12.
linking the formatted data to common sources of the source menu items represented by source identification data such that at least one food or beverage delivery service is linked to each common source and its source menu items;	See, above claim charts for independent claims 1 and 12. See, e.g., elements [1f], [5a] and [12f] in claim charts 1, 5 and 12.
identifying common menu items among the source menu items in the formatted data, and, for each identified common menu item, associating the source menu items with a master menu item;	See, above claim charts for independent claims 1 and 12. See, e.g., elements [1f] and [1g] in claim chart 1.
combining the linked data and the master menu items into a master data set; and	See, above claim charts for independent claims 1 and 12. See, e.g., elements [1h] and [12i] in claim charts 1 and 12.
importing the master data set and the source identification data into the searchable aggregated data structure.	See, above claim charts for independent claims 1 and 12. See, e.g., element [1i] in claim chart 1.

VP/#66313480

VP/#66313480